

An Integrated Event Management System for a University

Salman Khan*, Sushant Wankhade, Richa Bagdiya, Neehal Hussain, Rupali Deshmukh, Parminder Kaur

School of Engineering and Technology, MGM University, Chhatrapati Sambhajinagar, MS, India.

ABSTRACT

University technical festivals combine event discovery, registration, payment collection, venue attendance, certification, feedback, and reporting in one workflow, yet they are commonly handled through disconnected forms, spreadsheets, and payment dashboards. This paper presents the design, implementation, and functional verification of Supernova 2.0, a web-based event management platform deployed for the School of Engineering and Technology, MGM University, for a festival held on 4–5 September 2026 with nine flagship competitions. The system is a light-weight PHP/MySQL monolith that binds registration to server-side Razorpay payment verification, records physical venue attendance against the same participant identity, and gates certificate access on that attendance record. Administrative, organizer, and finance duties are separated through role-based access control, and a spreadsheet import path admits offline attendance records. Security controls include CSRF tokens, password hashing, session regeneration, login throttling, prepared statements, security headers, and HMAC signature verification. Validation is functional and workflow-oriented; no performance benchmark, usability study, or penetration test was performed, and no quantitative results are reported. The contribution is the integration of established mechanisms into a single working workflow in a focused, single-event institutional deployment.

Keywords: university event management, payment verification, attendance verification, certificate generation, role-based access control, Razorpay, PHP, MySQL

INTRODUCTION

A university technical festival behaves more like an information system than an event page. Organizers must publish competition details, capture participant data and payments, record attendance, maintain administrative records, and later link attendance to certificates and feedback. In practice these tasks are split across web forms, spreadsheets, payment dashboards, and manual attendance sheets, each holding its own view of who is attending what. Maintaining a single participant record from registration through to certificate therefore requires manual cross-referencing.

The literature reports a steady move toward digitization in institutional event management, with recurring themes of centralized registration, QR-based or digital attendance, automated certification, and role-based administration [1]–[14]. Because these functions are common, generic CRUD-style event platforms offer little novelty on their own. The question addressed here is narrower: how can one focused platform keep

participant identity, payment verification, venue attendance, certificate eligibility, feedback, and staff permissions consistent within a single-festival deployment, without unnecessary architectural complexity?

Supernova 2.0 was built to answer this for the School of Engineering and Technology, MGM University [15]. The contributions of this paper are: (1) an integrated flow linking registration to server-verified payment; (2) venue attendance logic tied to the same registration and transaction identity; (3) certificate access gated on recorded attendance; and (4) role-separated administration with a spreadsheet import path for offline records. These are scoped improvements for one deployment, not universal solutions.

II. RELATED WORK

Centralized administration. Pieloor et al. describe a PHP/MySQL platform combining registration, certification, and attendance [2], and Chaturvedi et al. centralize registration and participation tracking for a

Relevant conflicts of interest/financial disclosures: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

university [3]. Other campus systems similarly emphasize shared event records and administrative access [9], [10], [12], [14].

Registration and attendance. Salsabila et al. combine registration with e-ticketing and e-certification [4], Siahaan and Murtako add QR attendance [5], and Widiyanto et al. study dual-mode QR validation [8]. A broader review reports that attendance is commonly verified with QR codes and, in some studies, with biometrics, RFID, or geolocation [1]. Supernova 2.0 shares the aim of tying digital identity to physical presence, but its implemented check-in matches registration, participant email, event, and a payment reference; it is not QR-based.

Certification and usability. Hidayat and Gusman present a PHP/MySQL e-certificate portal with QR validation [6]. Hadiwiyaniti et al. evaluate a campus event system using the System Usability Scale [7]; the present work includes no comparable user study and makes no usability claims.

AI-assisted management. Some studies explore generic or AI-based event management [10], [11], [13]. A repository audit of Supernova 2.0 found no operational LLM, RAG, chatbot, machine-learning, or NLP-sentiment components; feedback is limited to 1–5 ratings and validated text. AI is treated only as future work.

III. RESEARCH GAP AND OBJECTIVES

Registration, attendance, certification, and dashboards are not novel in themselves [1]–[14]. The gap addressed is contextual: many studies examine individual modules or evaluate only prototypes [1], whereas this work examines integration decisions in a real technical-festival deployment. It should be read as an implementation and deployment study, not a claim that no similar architecture exists. The objectives are to:

- design a lightweight web workflow for a university technical festival;
- connect registration to server-side payment verification and persistent records;
- verify physical attendance using participant, event, and payment-related identity;
- gate certificate access on verified attendance rather than registration alone;

- separate administrative, organizer, and finance duties through role-based access control;
- ingest offline attendance data through spreadsheet import; and
- verify the workflows through functional inspection without unsupported quantitative claims.

IV. SYSTEM REQUIREMENTS

Requirements were derived from the festival's operations: public event information (FR1), participant registration with a unique identifier (FR2), server-side Razorpay order creation and checkout (FR3), payment signature verification with order-metadata reconciliation (FR4), venue attendance that prevents duplicate check-in (FR5), attendance-gated certificates (FR6), 1–5 star feedback with text (FR7), authenticated event and participant management (FR8), payment and revenue reporting (FR9), spreadsheet attendance import (FR10), email confirmation via PHPMailer/SMTP (FR11), and spreadsheet/PDF export (FR12). Multi-tenancy and multi-institution billing are out of scope. Non-functional requirements were treated as implementation constraints: relational consistency, rejection of malformed or duplicate submissions, protected privileged endpoints, no SQL statements built by string concatenation, and separation of public and staff functions. No formal benchmark for throughput, usability, or availability was defined.

V. SYSTEM ARCHITECTURE

Supernova 2.0 is a lightweight PHP monolith in an approximate MVC arrangement served by Apache. PDO handles database access; Vanilla JavaScript handles validation, filtering, asynchronous requests, checkout handoff, and document generation; Tailwind CSS handles styling. Razorpay and SMTP (via PHPMailer) are the external services. The design avoids microservices and cloud orchestration, which suits a single-event boundary. Fig. 1 shows the four logical layers: participant-facing pages; a shared security and routing layer (sessions, role checks, CSRF validation, headers, and throttling); PHP endpoints implementing business rules; and the MySQL/MariaDB persistence layer. Table I lists the stack.

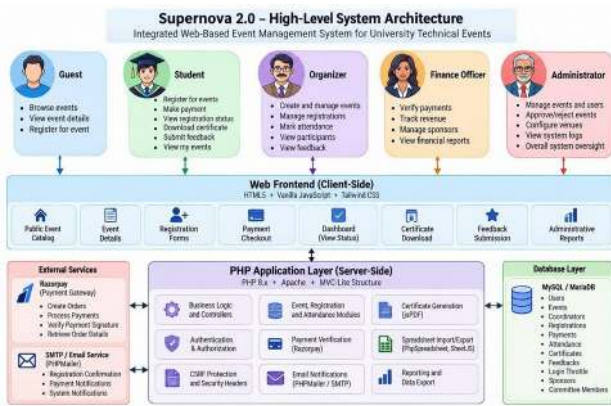


Figure 1: High-level System Architecture.

Reference/Explanation: This diagram presents the principal components of the Supernova 2.0 system, including user roles, the web frontend, the PHP application layer, external payment and email services, and the MySQL/MariaDB database. Source: Authors' own implementation.

Table I: Technology Stack

Layer	Technology	Role
Server	PHP 8.x; Apache	Requests, authentication, business logic
Database	MySQL/MariaDB; InnoDB	Event, participant, payment, attendance records
Data access	PDO	Parameterized queries
Frontend	HTML5; Vanilla JS; Tailwind CSS	Forms, filtering, checkout, scanner UI
Payments	Razorpay REST API	Order creation and payment flow
Email	PHPMailer; SMTP	Registration confirmation
Spreadsheet	PhpSpreadsheet; SheetJS	Import and export
PDF	jsPDF; AutoTable	Certificates and financial reports

Reference/Explanation: This table summarizes the technologies used to implement the Supernova 2.0 platform and maps each technology to its primary system role. Source: Authors' compilation based on the implemented system.

VI. DATABASE DESIGN

The relational schema follows the event lifecycle (Fig. 2). Participant data resides in *registrations*; staff credentials and roles in *users*; and further tables hold events, coordinators, payments, attendance, certificates, feedback, login-throttle state, sponsors, and committee members. The principal relationships link registrations to payments, attendance, and certificates, with events as the shared context. Each registration has a unique registration ID, and the pair (email, eventId) is unique, preventing duplicate registration for the same event while allowing one person to register for several events. Payment rows store Razorpay order and payment IDs, signature, amount, currency, and status; attendance rows store the registration/event association and a check-in timestamp; feedback rows store a rating constrained to 1–5 and text.

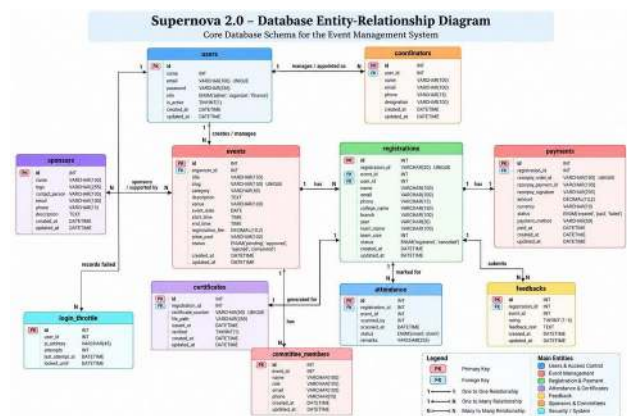


Figure 2: Database Entity-Relationship Diagram.

Reference/Explanation: This diagram represents the relational data model used to connect registrations with events, payments, attendance, certificates, feedback, and supporting administrative records. Source: Authors' own database design.

VII. SYSTEM IMPLEMENTATION

A. Event Management and Registration

The public catalog lists each competition with rules, venue, dates, reporting time, fees, prize pool, and coordinators; filtering runs in the browser over database-backed data. Administrators manage events and status through protected endpoints. Registration data is validated on the client and again on the server, which checks the phone number, email address, and

text fields and confirms that the (email, eventId) pair is unique. A registration ID with a Supernova-specific prefix is then generated. For paid events, registration is not finalized until payment is verified.

B. Payment Verification

The server creates the Razorpay order and stores participant and event identifiers in the order notes (Fig. 3). After checkout, the client returns the order ID, payment ID, and signature. The server recomputes the HMAC-SHA256 signature, compares it using `hash_equals()`, fetches the authoritative order from Razorpay, and reconciles its notes with the participant and event before committing registration and payment rows in a single database transaction. This moves the trust boundary from the browser to the server and mitigates client-side tampering; it does not prevent payment fraud in general.

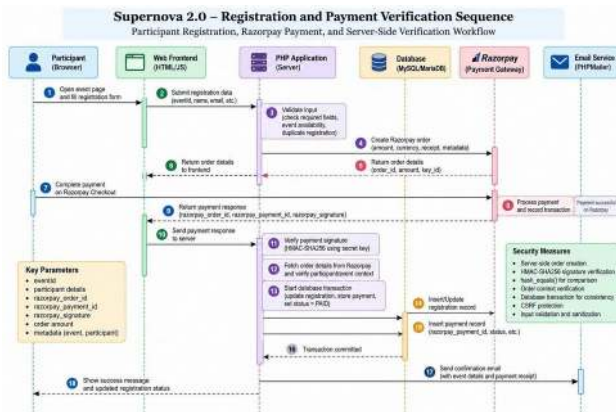


Figure 3: Registration and Payment Verification Sequence.

Reference/Explanation: This sequence illustrates the registration and payment workflow, including server-side Razorpay verification, metadata reconciliation, and transactional persistence of registration and payment records. Source: Authors' own implementation.

C. Attendance

Attendance is treated as a physical venue operation (Fig. 4). The organizer supplies the participant email, event, and transaction reference; the server checks the matching registration and payment context, rejects the request if an attendance row already exists, and otherwise inserts a timestamped record. A scanner-oriented page exists in the repository but disables remote web check-in; the verification primitive is server-side identity matching, not QR.

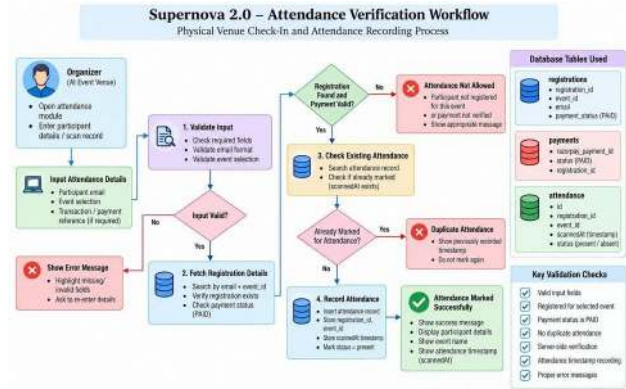


Figure 4: Attendance Verification Workflow.

Reference/Explanation: This workflow illustrates how participant, event, and transaction information is validated before attendance is recorded, while duplicate or inconsistent attendance requests are rejected. Source: Authors' own implementation.

D. Certificate Generation

The certificate portal requests a registration ID and registered email, retrieves the registration together with its attendance timestamp, and refuses the request if none exists. Otherwise it enables client-side PDF generation with jsPDF. Certificate access is therefore tied to recorded participation. The certificate is not cryptographically signed, and its integrity was not independently certified; database audit fields exist in the *certificates* table.

E. Feedback

Participants select an event, give a 1–5 rating, and add text (500-character maximum, with a minimum length for submission). Dashboards show the average rating and its distribution. No sentiment or NLP analysis is implemented.

F. Administrative and Financial Reporting

ADMIN users oversee events, participants, and aggregate registration, payment, attendance, and feedback data. ORGANIZER users manage rosters, attendance operations, and participant exports. FINANCE users access revenue, transaction status, payment audit views, and exports. Because records remain relational, registration, payment, attendance, and feedback can be reported together. PhpSpreadsheet handles server-side exports; SheetJS and jsPDF/AutoTable generate client-side spreadsheets and PDF reports.

G. Spreadsheet Integration

For attendance recorded outside the web workflow, PhpSpreadsheet parses .xlsx, .xls, and .csv files and recognizes headers such as Name, Email, Mobile, College, and Transaction ID. Missing participants are created as registrations with an EXCEL_IMPORT source marker plus associated attendance and payment rows. Imported data is only as reliable as the source sheet and its field mapping, and no parsing-throughput benchmark exists.

VIII. SECURITY AND DATA INTEGRITY

State-changing requests carry CSRF tokens generated from cryptographically secure random bytes and compared with `hash_equals()`. Passwords use `password_hash()`/`password_verify()` (bcrypt), and `session_regenerate_id(true)` is called at login to reduce session-fixation exposure. Role middleware gates privileged endpoints, reviewed SQL paths use PDO prepared statements, and responses carry X-Content-Type-Options, X-Frame-Options, Referrer-Policy, and Permissions-Policy headers. Login throttling applies a sliding window of five failed attempts per ten minutes per email/IP context. Table II summarizes the controls and Table III the role boundaries. These controls document implemented mechanisms, not measured attack resistance; the inspection also noted configuration-level hardening points (fallback credential handling, explicit session-cookie attributes) as maintenance items.

word_hash ()/password_verify () (bcrypt), and session_regenerate_id (true) is called at login to reduce session-fixation exposure. Role middleware gates privileged endpoints, reviewed SQL paths use PDO prepared statements, and responses carry X-Content-Type-Options, X-Frame-Options, Referrer-Policy, and Permissions-Policy headers. Login throttling applies a sliding window of five failed attempts per ten minutes per email/IP context. Table II summarizes the controls and Table III the role boundaries. These controls document implemented mechanisms, not measured attack resistance; the inspection also noted configuration-level hardening points (fallback credential handling, explicit session-cookie attributes) as maintenance items.

Table II: Implemented Security Controls

Control	Implementation	Status
CSRF protection	<code>random_bytes(32); hash_equals()</code>	Implemented
Password hashing	<code>password_hash / verify</code>	Implemented
Session regeneration	<code>session_regenerate_id(true)</code>	Implemented
Login throttling	5 attempts / 10 min	Logic verified in source
Parameterized SQL	PDO prepared statements	Implemented in reviewed paths
Security headers	<code>nosniff; SAMEORIGIN; Referrer/Permissions-Policy</code>	Implemented
Payment signature	<code>HMAC-SHA256; hash_equals()</code>	Implemented
Role-based access	ADMIN / ORGANIZER / FINANCE	Implemented

Reference/Explanation: This table summarizes the security mechanisms implemented in the application and their verification status based on the inspected

code paths. Source: Authors' security implementation and functional verification.

Table III: Role-based Access Control

Role	Core access	Boundary
Participant	Discovery, registration, feedback, attendance and certificate self-service	No staff controls
ADMIN	Event lifecycle, participant oversight, feedback analytics	Admin endpoints
ORGANIZER	Rosters, attendance, spreadsheet import, participant exports	No finance or system config
FINANCE	Payment ledger, revenue views, payment exports	No event administration

Reference/Explanation: This table summarizes the access responsibilities and boundaries assigned to participant, administrator, organizer, and finance roles. Source: Authors' own implementation.

IX. FUNCTIONAL VERIFICATION

Evaluation is limited to functional and workflow verification by code-path inspection. No formal benchmark, penetration test, usability study, or load test was performed. Table IV lists representative cases; each row shows that the condition and response exist in the inspected code, not that all runtime faults, browser variations, or attacks were tested.

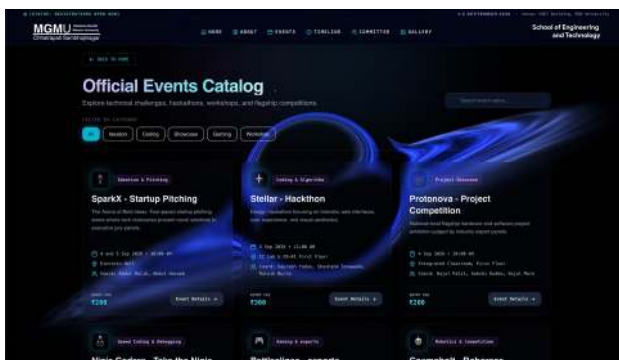
Table IV: Functional Verification Cases

Case	Condition	Expected behavior
Duplicate registration	Same email and eventId	Rejected (pre-check + unique key)
Valid signature	Correct IDs and signature	Payment accepted; records committed
Invalid signature	Modified signature	Verification fails
Context mismatch	Valid payment, inconsistent notes	Authoritative notes used before commit
Valid attendance	Matching registration, reference, email, event	Timestamp inserted
Duplicate attendance	Row already exists	Second check-in rejected
Invalid identity	Email or reference mismatch	Attendance not recorded
Certificate, no attendance	scannedAt null	Access refused
Certificate, attended	Attendance present	PDF generation enabled
Repeated failed login	5 failures in window	Login throttled
Unauthorized role	Missing role	Endpoint denied
Invalid CSRF	Missing/wrong token	Request rejected
Spreadsheet import	Recognized headers	Rows mapped to event records
Feedback validation	Rating outside 1–5 or short text	Submission rejected

Reference/Explanation: This table summarizes representative functional verification cases used to inspect the implemented workflow conditions and expected responses. Source: Authors' functional verification of the implemented system.

X. DEPLOYMENT: SUPERNOVA 2.0

The system was deployed for the MGM University SOET technical festival on 4–5 September 2026 at supernova.mgmu.ac.in, publishing the catalog, schedules, venues, terms, fees, and organizing-committee information [15]. Nine flagship competitions were offered: Battleclipse, Cosmobolt, Protonova, Junior Shark, Ninja Coders, SparkX, Infinity Lab, Stellar, and CodeBurst. The deployment is a working instance rather than a prototype with dummy data. It is intentionally single-event: there is no multi-tenancy, cross-festival payment configuration, or reusable branding, which keeps overhead low and focuses effort on registration integrity, venue verification, staff governance, and reporting. The available evidence does not include verified participant counts, payment totals, attendance totals, uptime, load profiles, or satisfaction scores, so none are reported.



DISCUSSION

Each module (registration, payments, attendance, certificates, and dashboards) is an established pattern; the contribution lies in keeping them connected. A registration carries a unique identifier and event context, a verified payment is attached to that record, attendance is recorded against the same identity, and certificate access depends on that attendance record. Finance and admin views read the same relational data rather than separate spreadsheets.

The payment path is the most consequential boundary. Server-side order creation, metadata reconciliation, signature validation, and transactional writes move trust away from the browser, though effectiveness was not independently tested. Attendance adds physical participation as a distinct verification state so that certificates do not follow automatically from payment; it still relies on correct venue procedure and accurate identity data at check-in. Spreadsheet import gives externally collected attendance a controlled route into the central database at the cost of a separate data-quality boundary.

The work shows that these stages can be connected on a conventional stack, but not how the system behaves under load, how usable it is, or how well its security controls resist attack. Reuse across festivals would require abstractions for festival identity, payment credentials, certificate templates, and branding; how much refactoring this needs remains unevaluated.

XII. LIMITATIONS AND FUTURE WORK

The system is a single-event platform and is not evaluated as multi-event or multi-tenant. Validation is functional, not quantitative: there is no load test, response-time study, longitudinal evaluation, or standardized usability study, and no independent penetration test. The platform depends on network connectivity and on Razorpay; synchronous SMTP confirmation may be delayed by mail-server latency; venue attendance depends on correct physical procedures; and spreadsheet import inherits the quality of its source data.

Future work includes generalizing to multi-event deployments with festival-level configuration, participant-specific QR attendance, a mobile client, richer notifications, and ERP/SIS interoperability. Standardized usability evaluation (e.g., SUS), performance benchmarking, multi-festival longitudinal deployment, and formal security assessment would provide stronger evidence. AI/NLP could later support participant assistance, feedback analysis, or predictive analytics, but is not part of the current implementation.

CONCLUSION

This paper presented Supernova 2.0, a web-based event management system deployed for the School of Engineering and Technology, MGM University. It unifies discovery, Razorpay-verified registration, physical attendance verification, attendance-gated certificates, feedback, role-specific dashboards, financial reporting, and spreadsheet import on a lightweight PHP/MySQL architecture. Functional inspection indicates that the key workflows contain the intended validity checks and state transitions. Performance, usability, security effectiveness, and adoption were not assessed and are left to future empirical work.

REFERENCES

1. SciSpace, "Web Based Event Management in Universities," systematic literature review report, 2026.
2. J. J. Pieloor, D. C. Parcasio, and M. L. Flores, "Centralized event management platform for digital transformation in continuing education," *Cognizance J.*, vol. 5, no. 10, pp. 433–448, 2025, doi: 10.47760/cognizance.2025.v05i10.041.
3. A. Chaturvedi, K. Sharma, A. Dua, and A. Gupta, "CU-Events: A comprehensive event management system for university," *Int. J. Res. Appl. Sci. Eng. Technol.*, 2024, doi: 10.22214/ijraset.2024.65020.
4. W. Salsabila, A. Taqwa, and L. Lindawati, "Rancangan aplikasi pengelolaan event, pemesanan e-tiket dan e-sertifikat di Politeknik Negeri Sriwijaya," *J. Media Inform. Budidarma*, vol. 6, no. 4, 2022, doi: 10.30865/mib.v6i4.4470.
5. T. D. A. Siahaan and A. Murtako, "Rancang bangun aplikasi event management berbasis web (studi kasus PT. Jasa Marga (Persero) Tbk.)," 2021, doi: 10.35814/jiac.v2i1.2689.
6. M. R. Hidayat and D. Gusman, "Portal e-Sertifikat dengan QR Code menggunakan PHP dan MySQL," 2022, doi: 10.31004/jerkin.v1i1.1.
7. R. Hadiwiyanti, T. L. M. Suryanto, and E. M. Safitri, "Evaluation of campus event management information system using system usability scale method," 2022, doi: 10.1109/ITIS57155.2022.10009978.
8. A. Widiyanto et al., "Implementasi sistem registrasi wisuda terintegrasi dengan validasi kehadiran dual-mode berbasis QR code," *KERNEL*, 2025, doi: 10.31284/j.kernel.2025.v6i2.8423.
9. J. R. V. Jeny, P. Sadhana, B. J. Kumar, S. Abhishek, and T. S. Chander, "A web based-college event management system and notification sender," 2022, doi: 10.1109/ICIRCA54612.2022.9985774.
10. S. Islam, R. Majumder, S. Sultana, S. Nasrin, and R. Islam, "Toward a generic event management system for academia," 2019, doi: 10.1109/ICAEE48663.2019.8975626.
11. G. Thombare, P. Jadhav, S. Dhere, P. Jadhav, and P. Raikar, "Evolving trends in college event management: A comprehensive survey and implementation analysis," *Int. J. Adv. Res. Sci. Commun. Technol.*, 2024, doi: 10.48175/ijarsct-17045.
12. C. L. T. Huyen, "Method and system for internet based event management at universities (case study: Van Lang University)," 2021, doi: 10.1063/5.0067064.
13. P. S. Hada, Yogesh, Bhupen, and Prince, "AI based event management web application,"

2022, doi: 10.1109/COM-IT-CON54601.2022.9850551.

14. A. Varma et al., "University event management web application by enhancing the coordination and engagement: 'uni'," 2024, doi: 10.69758/fjfk6725.
15. Supernova 2.0, "Official National Technical Fest Platform," MGM University, 2026. [Online]. Available: <https://supernova.mgmu.ac.in>. Accessed: Sep. 28, 2026. SUGGESTED PUBLICATION VENUES *Journals* IEEE Access is fast and open access, and it accepts applied system papers. It charges an article fee. *Education and Information Technologies* (Springer) and *Computer Applications in Engineering Education* (Wiley) are worth a look if you stress the university and education angle. Emerald and Elsevier information-systems journals are an option if you extend the paper with usability or performance results. Reviewers at stronger journals will expect this. *Upcoming*

conferences IEEE IC4 2027 (3rd IEEE International Conference on Computer, Communication and Control) is scheduled for Indore (hybrid), March 18–20, 2027. Its topic list includes computer science and engineering and information technology, so a systems paper fits. The listing is on an aggregator, so find the official IC4 site and check its deadline and IEEE status. InCIS 2027 (3rd India Conference on Information Systems) is hosted by IIT Madras in Chennai, March 12–14, 2027. It fits the topic well, but it is an information-systems research forum, so expect reviewers to want more evaluation than this paper has.

HOW TO CITE: Salman Khan, Sushant Wankhade, Richa Bagdiya, Neehal Hussain, Rupali Deshmukh, Parminder Kaur, An Integrated Event Management System for a University, Int. J. Sci. R. Tech., 2026, 3 (10), 1288-1295. <https://doi.org/10.5281/zenodo.23663423>